In the Specification

Please amend the specification of this application as follows:

Rewrite the paragraph at page 1, lines 6 to 9 as follows:

--This application claims priority to European Application Serial No. 00402331.3, filed August 21, 2000 ~~(TI-31366EU)~~ and to European Application Serial No. 00402945.0, filed October 24, 2000 ~~(TI-31359EU)~~. US Patent Application Serial No. ~~_____ (TI-31366US)~~ 09/932,651, now U.S. Patent No. 6,751,705, is incorporated herein by reference.--

Rewrite the paragraph at page 9, lines 10 to 27 as follows:

--Figure 4a illustrates a flow chart describing operation of a first embodiment of the power management tasks 38. In block 50, the power management tasks are invoked by the global scheduler 40, which could be executed on the MPU 12 or one of the DSPs 14; the scheduler ~~evaluate~~ evaluates the upcoming application and splits it into tasks with associated precedence and exclusion rules. The task list 52 could include, for example, audio/video decoding, display control, keyboard control, character recognition, and so on. In step 54, the task list 52 is evaluated in view of the task model file 56 and the accepted degradations file 58. The task model file 56 is part of the profiles 36 of the distributed applications layer 32. The task model file 56 is a previously generated file that assigns different models to each task in the task list. Each model is a collection of data, which could be derived experimentally or by computer aided software design techniques, which defines characteristics of the associated task, such as latency constraints, priority, data flows, initial energy estimate at a reference processor speed, impacts of degradations, and an execution profile on a given processor as a function of MIPs

and time. The degradation list 58 sets forth the variety of degradations that can be used in generating the scenario.--

Rewrite the paragraph at page 10, lines 1 to 23 as follows:

--Each time the task list is modified (i.e., a new task is created or a task is deleted) or when a real time event ~~occur,~~ occurs, based on the task list 52 and the task model 56 in step 54, a scenario is built. The scenario allocates the various tasks to the modules and provides priority information setting the priority with which tasks are executed. A scenario energy estimate 59 at a reference speed can be computed from the tasks' energy estimate. If necessary or desirable, tasks may be degraded; i.e., a mode of the task that uses fewer resources may be substituted for the full version of a task. From this scenario, an activities estimate is generated in block 60. The activities estimate uses task activity profiles 62 (from the profiling data 36 of the distributed application layer 32) and a hardware architectural model 64 (also from the profiling data 36 of the distributed application layer 32) to generate probabilistic values for hardware activities that will result from the scenario. The probabilistic values include each module's wait/run time share (effective MHz), accesses to caches and memories, I/O toggling rates and DMA flow requests and data volume. Using a period T that matches the thermal time constant, from the energy estimate 59 at a reference processor speed and the average activities derived in step 60 (particularly, effective processors speeds), it is possible to compute an average power dissipation that will be compared to thermal package model. If the power value exceeds any thresholds set forth in the package thermal model 72, the scenario is rejected in decision block 74. In this case, a new scenario is built in block 54 and steps 60, 66 and 70 are repeated. Otherwise, the scenario is used to execute the task list.--

Rewrite the paragraph at page 11, line 22 to page 12, line 5 as follows:

--Figure 4b is a flow chart describing operation of a second embodiment of the power management tasks 38. The flow of Figure 4b is the same as that of Figure ~~41,~~ 4a, except when the scenario construction algorithm is invoked (new task, task delete, real time event) in step 50, instead of choosing one new scenario, *n* different scenarios that match the performances constraints can be pre-computed in advance and stored in steps 54 and 59, in order to reduce the number of operations within the dynamic loop and provide faster adaptation if the power computed in the tracking loop leads to current scenario rejection in block 74. In Figure 4b, if the scenario is rejected, another pre-computed scenario is selected in block 65. Otherwise the operation is the same as shown in Figure 4a. --

Rewrite the paragraph at page 13, line 25 to page 14, line 6 as follows:

--The power compute block 66 is shown in Figure ~~8.~~ 7. In this block, the probabilistic activities from block 60 or the measured activities from block 76 are used to compute various energy values and, hence, power values over a period T. The power values are computed in association with hardware power profiles, which are specific to the hardware design of the multiprocessor system 10. The hardware profiles could include a Cpd for each module, logic design style (D-type flip-flop, latches, gated clocks and so on), supply voltages and capacitive loads on the outputs. Power computations can be made for integrated modules, and also for external memory or other external devices.--

Rewrite the paragraph at page 14, lines 16 to 26 as follows:

--Figure 9 illustrates and example of a multiprocessor system 10 using power/energy management software. In this example, the multiprocessor system 10 includes a MPU 12, executing an OS, and two DSPs 14 (individually referenced as DSP1 14a and DSP2 14b), each executing a respective RTOS. Each module is executing a monitor task 82, which monitors the values in various activity counters 78 throughout the multiprocessor system 10. The power compute task 84 is executed on DSP 14a. The various monitor tasks retrieve data from associated activity counters 78 and pass the information to DSP 14a to calculate a power value based on measured activities. The power management tasks, such as power compute task 84 and monitor task 82, can be executed along with other application tasks.--

Rewrite the paragraph at page 15, lines 1 to 3 as follows:

--In the preferred embodiment, the power management tasks 38 and profiles 36 (Figure 2) are implemented as JAVA class packages in a JAVA real-time environment.--

Rewrite the paragraph at page 16, lines 4 to 14 as follows:

--Figure 10 illustrates a portion of a processing system 10, showing a detailed block diagram of an autonomous processor (MPU 12), coupled to a coprocessor 16 along with other peripheral devices 100a and 100b. MPU 12 includes core circuitry 102, comprised of various core blocks 104a, 104b, and 104c. Core 102 further includes a Current Task ID register 106, a Task Priority register 108 and a Task Attributes register 110. Core 102 is coupled to a cache subsystem 112, including and instruction RAMset cache 114, a local RAM 116, an *n*-way instruction cache 118, an *n*-way data cache 120, a DMA (direct memory access) channel 122, and microTLB (translation lookaside buffer) caches ~~122a, 122b, and~~

~~122c.~~ 124a, 124b, and 124c. MPU 12 further includes voltage select circuitry ~~124~~ 126 for selecting between two (or more) voltages to power the MPU 12.--

Rewrite the paragraph at page 19, line 21 to page 20, line 2 as follows:

--Figures 13a and 13b illustrate the use of the Task Attributes register 110 to alter the configuration of the processing device 10 for more efficient operation. In this embodiment, the MPU Core 102 and Cache subsystem 112 are substantially the same as shown in Figures 10-12. A cache interface 130 couples the cache subsystem 112 to a traffic controller 132. Traffic controller ~~130~~ 132 and cache interface ~~132~~ 130 control the flow of traffic between the system buses and the components of the cache subsystem 112.--